



**VECTOR 5.0
EXECUTIVE**

User's Manual

VECTOR 5.3 EXECUTIVE
PROGRAM

USERS MANUAL
Revision A
January 29, 1982

P/N 7100-0250-00-00

Copyright 1982 Vector Graphic Inc.
Made in U.S.A.

Copyright 1982 by Vector Graphic Inc.
All rights reserved.

Disclaimer

Vector Graphic makes no representations or warranties with respect to the contents of this manual itself, even if the product it describes is covered by a warranty or repair agreement. Further, Vector Graphic reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation of Vector Graphic to notify any person of such revision or changes, except when an agreement to the contrary exists.

Revision Numbers

The date and revision of each page herein appears at the bottom of each page. The revision letter such as A or B changes if the manual has been improved but the product itself has not been significantly modified. The date and revision on the Title Page corresponds to that of the page most recently revised. When the product itself is modified significantly, the product will get a new revision number, as shown on the manual's title page, and the manual will revert to revision A, as if it were treating a brand new product. THIS MANUAL SHOULD ONLY BE USED WITH THE PRODUCT(S) IDENTIFIED ON THE TITLE PAGE.

TABLE OF CONTENTS

<u>Section</u>	<u>Page</u>
General Description.....	4
Table of Hex values.....	5
Command Format	
B - Jump to Bootstrap Loader-5-1/4" Floppy.....	6
C - Compare Blocks	
D - Dump in Hex	
E - External Communications	
F - Find Two Bytes	
G - Go to and Execute	
H - Display Memory Banks	
I - Input from a Port.....	7
J - CP/M Cold Boot	
K - Set Breakpoints	
L - Jump to 0000H	
M - Move Memory Block	
N - Non-destructive Memory Test	
O - Output to Port	
P - Program Memory.....	8
Q - Compute Checksum	
R - Register Dump	
S - Search for Single Byte	
T - Test Memory	
U - Jump to 0100H	
W - Jump to Bootstrap Loader-Winchester Hard Disk	
Y - Keyboard Echo	
Z - Zero or Fill Memory.....	9
Video Driver.....	9
Cursor X Y Positioning.....	10
Set Top of Screen.....	10
Keyboard Code Conversion for Vector Graphic Keyboards.....	10
Executive Listing.....	pages following

GENERAL DESCRIPTION

The Version 5.0 executive is a complete systems executive, designed to support the new Vector Extended CP/M Operating System. It also drives the Flashwriter II (80 X 24) video display board, and the Vector Graphic serial and parallel keyboards. Thus it is recommended for use with the Vector Mindless Terminal.

Because of the nature of the relationship between Extended CP/M and the 5.0 Executive, it is strongly suggested that any user writing conventional machine language programs use the facilities present in the operating system rather than those present in the Executive. It is further suggested that the user does not implement input and output commands directly to hardware devices. See the BIOS section of the Extended CP/M manual for information on how to interface to most I/O devices (including the keyboard and screen of the Mindless Terminal). See the BDOS section of the Extended CP/M manual to find out how to interface to the Disk Drive(s) in your system.

The 5.0 Executive differs so significantly from previous versions of the Extended Systems Monitor that it was given a different name. The changes made were necessary in order to accomodate the single/multiple user features of the Vector Operating System. In addition, disk boot driver routines have been included. Previously these were incorporated into a separate disk boot PROM.

This program includes an extensive command executive, a compactly written program designed to facilitate manipulation and display of memory data. The "prompt" which indicates that the Executive is waiting for operator entry is "Exec>".

If you are operating a terminal in a Multi-User system and are not quite sure what you are doing, be aware that Executive commands may have undesirable effects on other users of your system. It is suggested that you gain experience when there are no other active users on the system.

There are 23 commands which are entered as a single letter followed by up to four hexadecimal data fields. After each field is entered, a space is automatically output as a prompt. Either upper or lower case alpha characters may be used, but lower case characters will be converted to upper case, and any non-hex characters will be ignored. Allowable hex characters are 0-9, A-F. Address fields are four digits long; other fields are two digits long.

If a space is typed at any time during field entry, a default value of zero is assumed for all leading zeroes. This applies to an entire field as well as one that has been partially entered, and the cursor will advance to the next field if required. For example, typing (SP) will have the same effect as typing 0000; typing 100(SP) will have the same effect as 0100.

Any command that generates a display can be temporarily halted by depressing the space bar and continued by pressing the space bar again. The ESCape key will abort a display or command entry.

The 5.0 Executive is located at address E000H - EBFFFH in Vector Graphic systems. The physical implementation of this program may vary according to the system involved. In current systems as shipped from the factory, the Executive occupies the lower three quarters of the address space on a 2732 EPROM (or equivalent). The upper quarter of the address space on that EPROM is not used. The ZCB board in these systems has been modified to not respond to any memory address in the E000H to EFFFH range. This allows these addresses to be used by other memory boards in the system.

HEXADECIMAL NUMBERS

The hexadecimal number system may seem confusing if you are not familiar with it, but is clearly the best system with 16 bit addresses and 8 bit data. It is usually not necessary to convert between number systems, as this is usually done by software (i.e. assemblers). An explanation of hexadecimal and other number systems used in microcomputers may be found in virtually any introductory microcomputer book.

HEX NUMBER	DECIMAL VALUE	JARGON	BINARY BITS
0	0		1
1	1		1
2	2		2
A	10		4
B	11		4
C	12		4
D	13		4
E	14		4
F	15		4
10	16		5
FF	255		8
100	256	1 PAGE	9
3FF	1,023		10
400	1,024	1K	11
FFF	4,395		12
1000	4,096	4K	13
4000	16,384	16K	15
8000	32,768	32K	16
FFFF	65,535	64K-1	16

COMMAND FORMATExec>B - BOOT FLOPPY

Typing this command causes a jump to location E800H which is located in the disk boot section of the Executive. This will cause the disk operating system to be loaded into memory and transfer control to CP/M. This is designed to be used with a Vector system using the DualMode or FD controller board. The use of a Micropolis Disk Controller board is incompatible with this system.

Exec>C <ADR1> <ADR2> <ADR3> - COMPARE BLOCKS

A byte-by-byte comparison will be made between the block of memory data starting at ADR1 and ending at ADR2 and a block of identical length starting at ADR3. The differences will be printed out with the address, the byte in the first block and the byte in the second block. This command is useful to compare two versions of a program or to verify that proms have been programmed correctly.

Exec>D <ADR1> <ADR2> - DUMP IN HEX

Memory contents from ADR1 through ADR2 will be displayed as pairs of hexadecimal characters. The left character in each pair represents the four most significant bits of the memory location. The display may be halted and interrupted as described above. The ASCII representation is displayed in a column on the right.

Exec>E - EXTERNAL COMMUNICATIONS

The Executive will output anything typed on the keyboard through port 4 on the ZCB single board computer, the Bitstreamer II I/O board or an appropriately addressed Bitstreamer I board. Anything received on this port will be displayed on the screen. Normally a 300 baud modem would be connected to the serial RS-232 output from the I/O board, and this feature allows the system to be used as a simple terminal to communicate with a host in a full duplex mode. Operation at speeds above 300 baud requires the host to send null characters after linefeeds, so that characters are not lost when the screen scrolls up.

Exec>F <ADR1> <ADR2> <BYTE1> <BYTE2> - FIND TWO BYTES

This memory range from ADR1 through ADR2 will be searched for the particular code combination BYTE 1 BYTE 2. This is useful for locating particular commands or jump addresses. For example, if you wish to change a control character (say control D) in a program you may try FE 04, which is CPI 04 since this is a common way of testing input characters. If you wish to find all locations that call or jump to a particular address, say C700H, then search for 00C7. There is no guarantee that each location displayed is valid object code - it may be part of a data table, ASCII string, or second and third bytes of a three byte instruction.

Exec>G <ADR1> - GO TO AND EXECUTE

This command will cause a jump to ADR1 to execute a program or user subroutine. As with all Executive jump commands, the address contained on the stack is "START" (E04CR) and if the user routine at ADR1 ends in "RET", program execution will return to the Executive. Approximately 96 levels of stack space is available, but of course, pushing more registers on the stack than are popped will defeat the return feature with undesirable effects.

Exec>H - DISPLAY MEMORY BANKS

This command displays the bank number of resident video and Ram memory boards found in the system.

Exec>I <PORT> - INPUT FROM A PORT

Execution of this command will cause the CPU to execute an "IN PORT" instruction and the accumulator contents immediately following this to be displayed. This command is useful in checking out peripheral equipment. Only those ports used by the terminal, cassette interface, etc., will contain interesting values. All others will read FF since the data bus will be floating when the "IN" command is executed.

Exec>J - COLD BOOT

This command first checks to see which operating system is present in the system and then jumps to F800H. This will perform a cold boot of the operating system.

Exec>K - SET BREAKPOINTS

This command expects a 4 digit address, and will place a RESTART 7 (FF) at that location in RAM. When that instruction is executed, which is a call to location 0038H, the CPU will jump to the Executive routine that dumps the register contents. The instruction replaced with FF will also be restored. If a program is loaded over 0038H, the breakpoint instruction will be defeated unless RESET is depressed. Entry of the Executive at E000H will clear the breakpoint, as will pressing the RESET switch.

Exec>L - JUMP TO LOW RAM AT 0000H

This command jumps to memory location 0000H which is the beginning of program memory. This is the CP/M warm start location.

Exec>M <ADR1> <ADR2> <ADR3> - MOVE MEMORY BLOCK

The data contained in memory starting at ADR1 and ending at ADR2 is moved to memory locations starting at ADR3. This command is useful for moving a program from a temporary storage location to its correct address. If there is an overlap of the two memory areas, interesting results are obtained. For example, M 6000 7BFF 6400 will cause the block of data from 6000H through 63FFH to be repeated 8 times from 6000H through 7FFFH, since by the time location 6400H is read, it has been overwritten with data from 6000H. This is useful for bank programming of PROMs, or for creating repeating instruction sequences for test purposes.

Exec>N - NON-DESTRUCTIVE MEMORY TEST

Memory locations starting at 0000H are read and the data temporarily stored. The memory location is then tested to see if 00 and FF can be written and read correctly. This continues after rewriting the original data until the first error is detected, whereupon the address is displayed followed by the data written into memory and what was read from it. This command is most useful for checking how much memory a system contains. For example, if the system contains 16K of memory, 4000 00 FF should be printed, indicating that there is no memory at address 4000H. Since the test is non-destructive to data in memory, it can be used at any time.

Exec>O <PORT> <DATA> - OUTPUT TO PORT

The two hex digits "DATA" are loaded into the accumulator and the instruction "OUT PORT" is executed. This command is useful for checking out peripheral equipment. For example, if a printer is connected to I/O port 6, O 06 41 will cause an "A" to be printed since 41 is the hex ASCII code for "A". If there are other users on the system, be careful that you do not output to the port address of their memory boards as this may cause loss of data.

VECTOR GRAPHIC

Exec>P <ADR1> - PROGRAM MEMORY

The contents of 16 bytes of memory containing ADR1 are displayed in both hex and ASCII, allowing preceding and following instructions to be viewed. Advancing to the next instruction is accomplished by typing space or cursor right (right arrow). Backspace or cursor left (left arrow) goes backwards. The cursor up and down keys move to an adjacent 16 byte block. Any hex characters typed will replace the existing contents of RAM. After every keypress, the screen display is refreshed by reading from memory, so the display reflects the exact memory contents. To terminate, depress ESCAPE.

Exec>Q <ADR1> <ADR2> - COMPUTE CHECKSUM

The MOD 256 checksum of memory contents in the address range specified is computed and displayed. This command is useful for checking progs or files to see if anything has changed. Any source file or program written in pure code (it does not write on itself) will have the same checksum as when it was loaded. While debugging assembly language programs, it is useful to be able to verify that a program being debugged has not written garbage in the source file or assembler.

Exec>R - REGISTER DUMP

This command will print a header identifying the Z-80 registers, and immediately below it the contents of all the registers. The flags are displayed with the letters Z C M E H for the zero, carry, minus, parity even, and auxiliary or half carry flags respectively. The presence of the letter indicates the flag is true. The contents of the memory locations pointed to by the B, D, and H register pairs are also displayed as is the return address on the stack.

Exec>S <ADR1> <ADR2> <BYTE> - SEARCH FOR SINGLE BYTE

This is similar to the "F" command, except that only one byte is searched for instead of two. An example of the use of this command is to display all locations in a program where an output to a port occurs (D3). The address of each location will be displayed followed by "D3" and the next byte (the port number).

Exec>T <ADR1> <ADR2> - TEST MEMORY

This is an extremely useful command, especially when first setting up a system. This command permits thorough testing of the system memory. A portion of a 64K byte pseudorandom number sequence is written into memory from ADR1 through ADR2, and the exact same sequence is regenerated from the initial point and compared with what is read from memory. If all locations compare, another portion of the sequence is used to repeat the test which continues until it is interrupted. Any memory errors are displayed with the address, what was written into memory and what was read from memory, respectively. This information is all that is needed to pinpoint a malfunctioning memory chip. This test is quite exhaustive if used for at least 10 cycles and is far superior to incrementing or complementing tests which may not reveal addressing problems. The only area of system memory that cannot be tested with this routine is the few bytes required for the stack and video flags in the vicinity of FFD0H on the ZCB board. Do not use this test if there are other users active on the system.

Exec>U - JUMP TO 0100H

This command permits easy return to programs in the transient program area of CP/M.

Exec>W - WINCHESTER DRIVE BOOT

Typing this command will cause a jump to E802H which is in the Disk Boot section of the Executive and contains the Winchester drive bootstrap loader. The boot program will cause the CP/M operating system to be loaded into memory and control to be transferred to CP/M.

Exec>Y - KEYBOARD ECHO

This command causes keyboard input to be echoed directly to the video driver and can be used for demonstration purposes. An ESCape returns to the Executive.

Exec>Z <ADR1> <ADR2> <DATA> - ZERO OR FILL MEMORY

The memory block from ADR1 through ADR2 is filled with the byte "DATA". This is useful for setting memory to Zero. The end of a file or assembled program will stand out more clearly if memory is first zeroed. For test purposes, single instructions can be executed continuously so that bus waveforms are more easily interpreted. This is done by filling a block of memory with a repeated instruction sequence with a jump to the start of the block so that the program loops continuously. Be careful with this command if there are other users on the system.

VIDEO DRIVER

Version 5.0 of the Executive contains an elaborate video driver. The purpose of the video driver is to accept a stream of ASCII codes, and to write them into the screen memory in the proper place, interpreting certain non printing control codes in a special way. There are several entry points to the video driver. E009H is recommended. The character code to be printed must be in the A register. A CALL E009 will cause the character to be printed on the screen at the cursor position. All registers will be preserved.

Control codes are generated by the keyboard by holding the control (CTRL) key down while a letter key is pressed. Control codes have values between 0 and 31, and are 64 less than the codes for the corresponding upper case letters. To demonstrate the features of the video driver, type Y after the Executive prompt, and any keyboard generated code will be echoed to the video driver. The following control codes are interpreted as special functions, while all others are ignored:

Decimal Value	Hex Value	Control Code	Description
2	2	(^B)	HOME THE CURSOR
4	4	(^D)	CLEAR THE SCREEN AND HOME CURSOR
5	5	(^E)	DISPLAY THE CODE IN B REGISTER
8	8	(^H)	DESTRUCTIVE BACKSPACE (also BACKSPACE key)
9	9	(^I)	TAB OVER TO THE NEXT 8 MULTIPLE (also TAB)
10	A	(^J)	LINEFEED (also LF Key)
13	D	(^M)	CARRIAGE RETURN (also RETURN key)
14	E	(^N)	TOGGLE CURSOR
16	10	(^P)	CLEAR TO END OF SCREEN
17	11	(^Q)	CLEAR TO END OF LINE
18	12	(^R)	CURSOR DOWN
20	14	(^T)	TOGGLE REVERSE VIDEO
21	15	(^U)	CURSOR UP
23	17	(^W)	CURSOR LEFT
24	18	(^X)	CLEAR TO START OF LINE
26	1A	(^Z)	CURSOR RIGHT
-7	1B	ESC	CURSOR XY POSITION LEAD-IN or TOP OF SCREEN LEAD-IN

Experiment with the keys. There are special keys on the keyboard to generate some of the codes such as RETURN, TAB and linefeed (LF). If you are using the Vector Graphic Keyboard or Mindless Terminal, there are also keys for the cursor control and BACKSPACE. A few of the functions are not self explanatory. A Control D sets the reverse video flag to normal in addition to clearing the screen and homing the cursor. A Control T will then toggle the reverse video flag from normal to reverse and back without printing on the screen.

In some cases it is desirable to print the symbol for a control code on the screen. This can be done in assembly language programs by putting the code for the symbol in the B register and calling the video driver with Control E (05) in A. Enter the following machine code at FC00H and execute it to demonstrate this feature: 06 01 3E 05 04 CD 09 E0 CD 9C E0 C3 J2 FC

CURSOR X Y POSITIONING

Many programs utilize random X Y positioning of the cursor. This is done by outputting a three byte sequence to the video driver. The first code is ESC (1BH) followed by the desired X position and Y position in hex. The top left corner of the screen is 0, 0. The assembly language sequence 1B 40 08 would cause the cursor to move to line 8, character position 64 on the screen. To send the same sequence to the Executive via Microsoft Basic, the following statement would be used: "PRINT CHR\$(27);CHR\$(X+128);CHR\$(Y+128);", where X would equal 64 (40H) and Y would equal 08 (08H). Adding the value of 128 to X and Y in this example sets the eighth bit high. This is done to avoid Microsoft Basic from confusing the values as control codes. This may not be demonstrated using the keyboard since ESC causes a return to the Executive.

The video driver provides an extensive range of special controls, however, they must be incorporated into the software generating the video stream to be meaningful. For instance a piece of software that merely echoes all characters as they go into its input buffer will allow cursor motion on the screen, but this will probably be meaningless to the software.

SETTING TOP OF SCREEN

The logical top of screen can be set by sending the appropriate codes (escape sequences) to the Executive program. To set the top of screen send: ESC DEL (line number) to the Video driver. The line number must be expressed in hexadecimal in the range of 0H to 16H (0 to 22 decimal).

KEYBOARD CODE CONVERSION - VECTOR GRAPHIC KEYBOARDS

Due to limitations in the keyboard encoder chip, the [] key on Vector Graphic keyboards is not encoded properly. The correct code is generated by a conversion routine in the Executive's CONVERT routine. The codes for backslash and tilde are also produced by the control and control shift mode of this key.

[] KEY CONVERSION:

MODE	KEYCODE	CONVERTED CODE	ASCII SYMBOL
unshifted	F1	5B	[
shifted	E1	5D]
control	B1	5C	\
control shift	A1	7E	-

The cursor up key is also converted from 60H to 15H which is interpreted correctly by the video driver. Room is provided in the routine for up to 15 keycode conversions. Foreign languages require additional conversions. It is essential that software utilize the Executive conversion routine for this reason.

E04C 7E	MOV	A,M	GET BYTE FROM MEMORY
E04D 35	LCR	M	CHANGE RAM
E04E BE	CMP	M	CHECK IF SAME
E04F 77	MOV	M,A	RESTORE RAM
E050 2811	JR2	USERCHECK20	SKIP IF NOT ENABLED
E052 22EAFB	SHLD	XYFLAG	ZEROS FLAGS (ORDER DEPENDENT)
E055 1A	LDAX	D	GET RAM FLAG
E056 B0	ORA	B	SET BANK BIT
E057 12	STAX	D	SAVE NEW RAM FLAG
E058 30C3	MVI	A,BC3H	JUMP
E05A 322800	STA	28H	RST 5
E05D 21D7E6	LXI	H,DUMPREGS	JUMP ADDRESS FOR RESTART
E060 222900	SHLD	29H	SAVE IN MEMORY AT RST 5
E063 2100F0	LXI	H,SCREEN	CHECK VIDEO RAM
E066 7E	MOV	A,M	GET BYTE FROM MEMORY
E067 35	DCR	M	CHANGE RAM
E068 BE	CMP	M	CHECK IF SAME
E069 77	MOV	M,A	RESTORE RAM
E06A 2812	JR2	USERCHECK30	SKIP IF NOT ENABLED
E06C 1B	DCX	D	(DE) = VIDEO FLAG
E06D 1A	LDAX	D	
E06E B0	ORA	B	SET BIT
E06F 12	STAX	D	SAVE NEW VIDEO FLAG
E070 13	INX	D	(DE) = RAMFLAG
E071 1A	LDAX	D	GET RAM FLAGS
E072 A8	ANA	B	MASK BANK WITH RAM BITS
E073 2809	JR2	USERCHECK30	SKIP IF NOT BOTH VIDEO AND RAM
E075 D9	EXX		SAVE ALL REGS
E076 CDFEE0	CALL	SIGN.ON	DISPLAY SIGN-ON
E079 AF	XRA	A	USE PORT #
E07A CD1BE0	CALL	INILOOP	INITIALIZES KEYBOARD PORT
E07D D9	EXX		RETRIEVE ALL REGS
E07E CB10	RALR	B	ROTATE BIT UP
E080 30C5	JRNC	USERCHECK10	LOOPS THRU ALL BANKS
E082 CB10	RALR	B	
E084 ED41	OUTP	B	TURN ON BANK 1
E086 3E8E	MVI	A,14	
E088 CD10E4	CALL	VIDEO	TOGGLE CURSOR ON
E088			
E088 2AE7FB	CLRBRK		
E08E 11E9FB	SHLD	BRKPTLOC	HIL = ADDRESS OF BREAKPOINT
E091 ED53E7FB	SDED	D,BRKCODE	DE = INSTRUCTION STORAGE
E095 1A	LDAX	D	SAVE DE AS BREAKPOINT ADDRESS
E096 77	MOV	M,A	GET INSTRUCTION
E097			PUT BACK IN MEMORY
E097 31D0FB	START	IXI	INITIALIZE STACK
E09A 2100F0	IXI	H,SCREEN	INITIALIZE TOP OF SCREEN
E09D 22DFFB	SHLD	TOSCN	
E0A0 CD82E5	KEYPOL	CALL	PROMPT
E0A3 CD02E2	KEYPOL	CALL	ESCAPE
E0A6 28FB	KEYPOL	JR2	READ KEYBOARD
E0A8 E65F	ANI	5FH	LOOP IF NO INPUT
E0AA 2197E0	LXI	H,START	CHANGE TO UPPER CASE
E0AD E5	PUSHI	H	PUSH RETURN ADDRESS
E0AE FE04	CPI	'D'-64	CHECK FOR CLRSON
E0B0 CC10E4	CZ	VIDEO	ECHO CLRSON
E0B3 FE41	CPI	'A'	RANGE CHECK

E0B5 D8	RC	050H	/TOO SMALL	
E0B6 FE58	CPI			
E0B8 D8	RNC		/TOO LARGE	
E0B9 21CAE0	LXI	H,CMDTB	;HIL = START OF COMMAND TABLE	
E0BC F5	PUSH	PSW	SAVE CHARACTER	
E0BD D641	SUI	'A'	SUBTRACT ASCII BIAS	
E0B9 87	ADD	A	DOUBLE FOR 2 BYTE ADDRESS	
E0C0 5F	MOV	E,A		
E0C1 1600	MVI	D,B	DE = INDEX	
E0C3 19	DAD	D	; (HL) = JUMP ADDRESS	
E0C4 5E	MOV	E,M	GET LSBYTE	
E0C5 23	INX	H		
E0C6 56	MOV	D,M	GET MSBYTE	
E0C7 EB	XCHG		;HL = JUMP ADDRESS	
E0C8 F1	POP	PSW		
E0C9 E9	PCHL		AWAY WE GO	
E0CA				
E0CA			COMMAND TABLE	
E0CA				
E0CA 97E0	CMDTB	DW	START	;A
E0CC 00E8		DW	FLBOOT	;B
E0CE 83E3		DW	COMPR	;C
E0D0 CBE5		DW	HEXRUL	;D
E0D2 DCE7		DW	EXTCOM	;E
E0D4 A2E3		DW	FIND	;F
E0D6 5DE1		DW	EXEC	;G
E0D8 68E1		DW	SYSTAT	;H
E0DA EEE3		DW	PLNPT	;I
E0DC 56E2		DW	COLD	;J
E0DE C1E7		DW	SETBRK	;K
E0E0 14E3		DW	LORM	;L
E0E2 39E3		DW	MOVEB	;M
E0E4 61E3		DW	NDMT	;N
E0E6 FDE3		DW	POUTP	;O
E0E8 14E6		DW	PROGRAM	;P
E0EA 3BE2		DW	CLKSM	;Q
E0EC CBE6		DW	DRGCS	;R
E0EE AFE3		DW	SNCII	;S
E0F0 87E2		DW	TMEM	;T
E0F2 0BE3		DW	USER	;U
E0F4 97E0		DW	START	;V
E0F6 02E8		DW	MSIX00T	;W
E0F8 97E0		DW	STARTT	;X
E0FA 77E2		DW	ECHO	;Y
E0FC 20E3		DW	ZEROM	;Z
E0FE				
E0FE 3E04	SIGN.ON:	MVI	A,4	
E100 CD10E4		CALL	VIDEO	
E103				
E103 21A3F1		LXI	H,HORIZ*5+SCREEN+19	
E106 11072A		LXI	D,2A07H	
E109 3E7F		MVI	A,07FH	
E10B CD4CE1		CALL	DRBOX	
E10E				
E10E 21F5F1		LXI	H,HORIZ*6+SCREEN+21	
E111 110526		LXI	D,2605H	
E114 3EA0		MVI	A,0A0H	

```

E116 CD4CE1      CALL  DRBOX
E119
E119 2147F2      LXI  H,HORIZ*7;SCREEN+23
E11C 110322      LXI  D,2203H
E11F 3E20      MVI  A,' '
E121 CD4CE1      CALL  DRBOX
E124
E124 CD77E5      CALL  PTSTRG
E127 1B1A08      DB   ESCP,26,8
E12A 56454354      DT   'VECTOR GRAPHIC EXECUTIVE 5.0b'
E12E 4F522047
E132 52415048
E136 494J2045
E13A 58454355
E13B 54495645
E142 20352E30
E146 62
E147 1B000D8E      DB   ESCP,0,13,14180H
E14B C9      RET
E14C
E14C E5      DRBOX: PUSH  H
E14D 42      MOV   B,D
E14E 77      DRBOXIN: MOV   M,A
E14F 23      THX  H
E150 10FC      DJNZ  DRBOX10
E152 E1      POP   H
E153 D5      PUSH  D
E154 115000      LXI  D,80
E157 19      DAD  D
E158 D1      POP   D
E159 1D      DCR  E
E15A 20F0      JRNZ  DRBOX
E15C C9      RET
E15D
E15D      /* EXECUTE THE PROGRAM AT THE ADDRESS */
E15D      /* */
E15D CD77E5      EXEC  CALL  PTSTRG
E160 474FA0      DHII 'GO '
E163 CD92E1      CALL  AIIEX      ;READ ADD FROM KB
E166 EB      XCIG
E167 E9      PCIIH      ;JUMP TO IT
E168
E168      /* DISPLAY SYSTEM HARDWARE STATUS
E168      /* */
E168 CD74E5      SYSTAT: CALL  RPPTSTRG
E168 5241CD      DHII 'RAM'
E16E 3AFFFF      LDA   RAMFLAG      ;POINT TO LOW RAM FLAG
E171 CD7FE1      CALL  CHECKUSER      ;CHECK AND PRINT ACTIVE
E174 CD74E5      CALL  RPPTSTRG      ;BANKS
E177 56494445      DHII 'VIDEO'
E17B CF
E17C 3AFFFF      LDA   VIDEOFLAG      ;NOW CHECK VIDEO FLAG
E17F 0608      MVI  B,8      ;TEST 8 BANKS
E181 1F      CHECKUSER: RAR      ;ROTATE BIT INTO CARRY
E182 300B      JRNC  CHECKUSER10      ;SKIP IF NOT SET
E184 4F      MOV   C,A      ;SAVE BANK BYTE
E185 CD4CE1      CALL  SPCE      ;PRINT SPACE

```

```

PAGE 6
E188 3E09      MVI  A,9      ;FORM BANK NUMBER
E18A 98      SUB  B
E18B CD4AE2      CALL  PT2      ;PRINT BANK NUMBER
E18E 79      MOV  A,C      ;RETRIEVE BANK BYTE
E18F 10F0      CHECKUSER10: DJNZ  CHECKUSER10      ;LOOP THRU ALL BANKS
E191 C9      RET
E192
E192      /* CONVERT UP TO 4 HEX DIGITS TO BIN
E192      */
E192 0E04      AIIEX      MVI  C,4      ;COUNT OF 4 DIGITS
E194 210000      AIIE0      LXI  H,8      ;16 BIT ZERO
E197 CD02F2      AIIE1      CALL  ESCAPE
E19A FE2B      CPI  ' '
E19C 281D      JRZ  SPCUVH
E19E CD08E1      CALL  HEX
E1A1 38F4      JRC  AIIE1      ;CHECK VALUE
E1A3 29      DAD  H      ;MULT H*16
E1A4 29      DAD  H
E1A5 29      DAD  H
E1A6 29      DAD  H
E1A7 85      ADD  L
E1A8 6F      MOV  L,A
E1A9 80      DCR  C      ;4 DIGITS?
E1AA 20E0      JRNZ  AIIE1      ;KEEP READING
E1AC EB      XCIG
E1AD 3E2B      SPCB      MVI  A,1,1      ;PRINT SPACE
E1AF C310E4      PTCH      JMP  VIDEO
E1B2 3E0D      CRLF      MVI  A,0DH
E1B4 CD10E4      CALL  VIDEO
E1B7 3E0A      MVI  A,0AH
E1B9 10F4      JR  PTCH
E1BB
E1BB 3D10E4      SPOVH      CALL  VIDEO
E1BE 18EC      JR  SPCB-1
E1C0
E1C0      /* CHECK FOR HEX VALUE, CONVERT
E1C0  FE30      HEX      CPI  '0'      ;<0
E1C2 D8      RC      CPI  '1'      ;>9
E1C3 FE3A      CPI  '2'      ;>9
E1C5 3009      JNC  NUM
E1C7 E65F      ANI  5FH      ;UPPER & LOWER CASE
E1C9 FE41      CPI  'A'      ;>A
E1C9 D8      RC
E1CC FE47      CPI  'G'      ;>F
E1CE 3F      CHC
E1CF D8      RC
E1D0 CD10E4      NUM      CALL  VIDEO
E1D1 D630      SUI  48      ;ASCII BIAS
E1D5 FE0A      CPI  10      ;DIGIT 0-10
E1D7 3002      JRC  A1,FA
E1D9 D607      SUI  7      ;ALPHA BIAS
E1D8 A7      ALFA      ANA  A      ;CLEAR CY
E1DC C9      RET      ;WITH CY CLEAR
E1DD
E1DD      /* READ 2 DIGITS FROM THE CONSOLE
E1DD  0E02      AIIE2      MVI  C,2
E1DF 18B3      JR  AIIE0

```

```

1E1 ; SHORT ROUTINE TO SAVE CODE
1E1 ; TAIEX CALL AJEX
1E1 CD92E1 JR AJEX
1E4 18AC
1E6 ;
1E6 ;** READ FROM CONSOLE TO REG A ***
1E6
1E6 CD02E2 RXCN CALL ESCAPE ;READ KEYBOARD
1E9 28FB
1EB FE60 CPI 60H
1ED 38C0 JAC PTCN
1EF E65F ANI 5F11
1F1 18BC JR PTCN
1F3 ;PAUSE CALL ESCAPE
1F3 CD02E2 CPI '
1F6 FE20 RINZ
1F8 C0
1F9 CD02E2 PLOOP CALL ESCAPE
1FC FE20 CPI '
1FE C2F9E1 JNZ PLOOP
201 C9 RET
202 ;ESCAPE CALL KEYSTAT
205 C8 RZ
206 CD17E2 CALL DATACONV
209 FE1B CPI ESCP ;ESCAPE
208 CA97E8 JZ START
20E C9 RET
20F ;KEYSTAT IN OONS
211 E640 ANI RDA
213 C9 RET
214 ; KEYBOARD DATA FETCH
214 DB01 KEYDATA: IN COND ;KEYBOARD DATA
216 C9 RET
217 ; KEYBOARD FETCH AND CODE CONVERSION
217
217 DB01 DATACONV: IN COND
219 E5 CONVERT: PUSH H
21A C5 PUSH B
21B 010500 LXI B, TABLEND-KTABL/2
21E 2131E2 LXI H, KTABLE
221 ED01 LOOP CCI ;COMPARE TABLE
223 2006 JRZ END
225 23 INX H
226 EA21E2 JPC LOOP ;CONT LOOKING
229 1801 JR NEND
22B 7E END MOV A, M ;NEW CODE
22C E67F NEND ANI 7FH ;MASK DOWN
22E C1 POP B
22F E1 POP H
230 C9 RET
231 ; THIS TABLE CAN BE EXTENDED IF DESIRED

E231 E15D KTABL DD 0E15DH ;)
E233 F15B DD 0F15BH ;)
E235 A17E DD 0A17EH ;-
E237 B15C DD 0B15CH ;\
E239 6015 DD 06015H ;CURSOR UP
E23B E23B = TABLEND EQU $ ORG KTABL+30 ;ROOM FOR 15 CONVS
E23B ; CHECKSUM ROUTINE
E23B CD77E5 CIKSM CALL PTSTNG
E23E 43484053 DTH 'CIKSM'
E242 554DA0
E245 CDE1E1 CALL TAIEX
E248 0600 MVI B, B
E24A 7E CHKSHLP MOV A, M
E24B 80 ADD B
E24C 47 MOV B, A
E24D CD803E3 CALL BMP
E250 20FB JNZ CHKSHLP
E252 78 MOV A, B
E253 C3EA82 JMP PT2
E256 ; CP/M COLD BOOT
E256 ; COLD CALL PTSTNG
E256 CD77E5 COLD DTH 'COLD BOOT'
E259 434F4C44
E25D 20424F4F
E261 D4
E262 3A00FB LDA COLDSTART
E265 FEC3 CPI 0C3H
E267 CA00FB JZ COLDSTART
E26A CD74E5 CALL RPTSTNG
E26D 4E4F2053 DTH 'NO SYSTEM'
E271 59535445
E275 CD
E276 C9 RET
E277 ; KEYBOARD ECHO ROUTINE
E277 ECHO CALL PTSTNG
E277 CD77E5 DTH 'ECHO'
E27A 4543484F
E27E A0
E27F CD02E2 ECOLP CALL ESCAPE ;LOOK AT KEYBOARD
E282 C410E4 CNZ VIDEO ;PRINT IF KEYPRESS
E285 18FB JR ECOLP ;CONTINUE LOOPING
E287 ;** MEMORY TEST ROUTINE ***
E287 ; THEM CALL PTSTNG
E287 CD77E5 DTH 'TEST'
E28A 54455354
E28E A0
E28F CDE1E1 CALL TAIEX
E292 015ASA LXI B, 5A5AH ;INI B,C
E295 CDC1E2 CYCL CALL RNDM
E298 C5 PUSH B ;READ ADDRESSES
E299 E5 PUSH H ;KEEP ALL REGS
E29A D5 PUSH D

```

E29B CDC1E2	TLOP	CALL RNDM		
E29E 78		MOV M,B	;WRITE IN MEM	
E29F C043E3		CALL BMP		
E2A2 C29BE2		JNZ TLOP	;REPEAT LOOP	
E2A5 D1		POP D		
E2A6 E1		POP H	;RESTORE ORIG	
E2A7 C1		POP B	;VALUES OF	
E2A8 E5		PUSH H		
E2A9 D5		PUSH D		
E2AA CDC1E2	RLOP	CALL RNDM	;GEN NEW SEQ	
E2AD 7E		MOV A,M	;READ MEM	
E2AE B8		CMP B	;COMP MEM	
E2AF C4E1E2		CNZ ERR	;CALL ERROR RTN	
E2B2 C103E3		CALL BMP		
E2B5 C2AAE2		JNZ RLOP		
E2B8 D1		POP D		
E2B9 E1		POP H		
E2BA 3E2E		MVI A,1.		
E2BC CD10E4		CALL VIDEO		
E2BF 18D4		JR CYCL		
E2C1		;** THIS ROUTINE GENERATES RANDOM NOS ***		
E2C1 CDF3E1	RNDM	CALL PAUSE		
E2C4 78		MOV A,B	;LOOK AT B	
E2C5 E684		ANI 0B4H	;MASK BITS	
E2C7 A7		ANA A	;CLEAR CY	
E2C8 EACCE2		JPE PEVE	;JUMP IF EVEN	
E2C9 37		STC		
E2CC 79	PEVE	MOV A,C	;LOOK AT C	
E2CD 17		RAL	;ROTATE CY IN	
E2CE 4F		MOV C,A	;RESTORE C	
E2CF 78		MOV A,B	;LOOK AT B	
E2D0 17		RAL	;ROTATE CY IN	
E2D1 47		MOV B,A	;RESTORE B	
E2D2 C9		RET	;RETURN W NEW B,C	
E2D3		;		
E2D3		;** ERROR PRINT OUT ROUTINE		
E2D3		;		
E2D3 CDB2E1	PTAD	CALL CHLP	;PRINT CR,LF	
E2D6 CDF3E1		CALI PAUSE		
E2D9 7C		MOV A,II	;PRINT	
E2DA CDEAE2		CALL PT2	;ASCII	
E2DD 7D		MOV A,L	;CODES	
E2DE C32BE7		JMP PT2S	;FOR ADDRESS	
E2E1		;		
E2E1 F5	EUR	PUSH PSW	;SAVE ACC	
E2E2 CDD3E2		CALL PTAD	;PRINT ADD.	
E2E5 78		MOV A,B	;DATA	
E2E6 CD2BE7		CALL PT2S	;WRITTEN	
E2E9 F1		POP PSW	;DATA READ	
E2EA F5	PT2	PUSH PSW		
E2EB CDF1E2		CALL BINH		
E2EE F1		POP PSW		
E2EF 18D4		JR BINL		
E2F1 1F	BINL	RAR	;SHIFT RLT 4 BITS	
E2F2 1F		RAR		
E2F3 1F		RAR		
E2F4 1F		RAR		

E2F5 E60F	BINL	ANI 0FH	;LOW 4 BITS
E2F7 C630		ADI 48	;ASCII BIAS
E2F9 FE3A		CPI 58	;DIGIT 0-9
E2FB DA10E4		JC VIDEO	
E2FE C607		ADI 7	;DIGIT A-F
E300 C310E4		JMP VIDEO	
E303		;	
E303 7B		; COMPARE ADDRESSES AND INCREMENT H	
E304 95	BMP	MOV A,E	
E305 2002		SUB L	
E307 7A		JNZ COON	
E308 9C		MOV A,D	
E309 23	COON	SBB H	
E30A C9		INX H	
E30B		RET	
E30B		;	
E30B CD77E5	USER	CALL PTSTNG	
E30B 5450C1		DTH 'TPA'	
E311 C30001		JMP TPA	
E314		;	
E314 CD77E5	LORAM	CALL PTSTNG	
E317 4C4F2052		DTH 'LO RAM'	
E31B 41CD		;	
E31D C30000		JMP H	
E320		;	
E320 CD77E5	ZERON	CALL PTSTNG	
E323 46494C4C		DTH 'FILL'	
E327 A8		;	
E328 CDE1E1		CALL TAHEx	;READ ADDRESSES
E32B E5		PUSH H	;SAVE H
E32C CDD0E1		CALL AHEx2	;READ 2 DIGITS
E32F EB		XCHG	
E330 E3		XTML	;RESTORE H,I.
E331 C1		POP B	
E332 71	ZLOOP	MOV M,C	;WRITE INTO MEM
E333 C043E3		CALL BMP	;COMP ADD., INCR H
E336 C8		RZ	;RETURN IF DONE
E337 18F9		JR ZLOOP	;CONTINUE TIL DONE
E339		;	
E339 47	MOVEB	MOV B,A	;SAVE CODE
E33A CD77E5		CALL PTSTNG	
E33D 4D4F5645		DTH 'MOVE'	
E341 A8		;	
E342 CDE1E1	MOVENTR	CALL TAHEx	;READ ADDRESSES
E345 E5		PUSH H	
E346 CD92E1		CALL AHEx	
E349 EB		XCHG	
E34A E3		XTML	
E34B 4B	MLDOP	MOV C,M	;BACK TO NORMAL.
E34C E3		XTML	
E34D 78		MOV A,B	
E34E FE4D		CPI 'M'	
E350 2004		JRZ NEXTH	
E352 7B		MOV A,M	

E353 E3 XTHL
 E354 77 MOV H,A
 E355 E3 XTHL
 E356 71 MOV H,C
 E357 23 INX H
 E358 E3 XTHL
 E359 CD00E3 CALL BMP
 E35C CA97E0 JZ START
 E35F 18EA JR MLOOP
 E361 ; NON DESTRUCTIVE MEMORY TEST
 E361 CD77E5 NDHT CALL PTSTNG
 E364 4D454D20 DTII 'MEM TOP'
 E368 541FD8
 E368 210000 LXI H,0 ;START AT ZERO
 E36E 4E NDLOP MOV C,M
 E36F 06FF MVI B,0FFH
 E371 70 MOV M,B
 E372 7E MOV A,M
 E373 B8 CMP B
 E374 C27CE3 JNZ ERJUP ;PRINT ERROR
 E377 0600 MVI B,B
 E379 70 MOV M,B
 E37A 7E MOV A,M
 E37B B8 CMP B
 E37C C2E1E2 ERJUP JNZ ERR
 E37F 71 MOV M,C
 E380 23 INX H
 E381 18EB JR NDLOP
 E383 ; COMPARE TWO BLOCKS OF MEMORY
 E383 CD77E5 COMPR CALL PTSTNG
 E386 434D50A0 DTII 'CMP'
 E38A CDE1E1 CALL TAHEX
 E38D E5 PUSH H
 E38E CD92E1 CALL AHEX
 E391 EB XCING
 E392 7E VMLOP MOV A,M
 E393 23 INX H
 E394 E3 XTHL
 E395 B8 CMP M
 E396 46 MOV B,M
 E397 C4E1E2 CNZ ERR
 E39A CD03E3 CALL BMP
 E39D E3 XTHL
 E39E 20F2 JRNZ VMLOP
 E3A0 P1 POP PSW
 E3A1 C9 RET
 E3A2 ; SEARCH FOR SPECIFIC CODES
 E3A2 F5 FIND PUSH PSW
 E3A3 CD77E5 CALL PTSTNG
 E3A6 46494E44 DTII 'FIND-2'
 E3A8 2D32A8
 E3AD 1800 JR SRCHNT
 E3AF F5 PUSH PSW
 E3B0 CD77E5 CALL PTSTNG
 E3B3 46494E44 DTII 'FIND-1'
 E3B7 2D31A8
 E3BA CDE1E1 SRCHNT CALL TAHEX

E3B0 E5 PUSH H ;SAVE H
 E3B8 C0DDE1 CALL AIE2 ;READ 2 DIGITS
 E3C1 EB XCING ;H=CODE, D=F
 E3C2 45 MOV B,L ;PUT CODE IN B
 E3C3 E1 POP H ;RESTORE H
 E3C4 F1 POP PSW
 E3C5 FE53 CPI 'S'
 E3C7 F5 PUSH PSW
 E3C8 2807 JRZ CONT
 E3CA E5 PUSH H
 E3CB C0DDE1 CALL AIE2 ;READ 2 DIGITS
 E3CE EB XCING
 E3CF 4D MOV C,L
 E3D0 E1 POP H
 E3D1 7E CONT MOV A,M ;READ MEMORY
 E3D2 B8 CMP B ;COMPARE TO CODE
 E3D3 2012 JRNZ SKP ;SKIP IF NO COMP
 E3D5 F1 POP PSW ;FETCH CONTROL
 E3D6 FE53 CPI 'S'
 E3D8 F5 PUSH PSW
 E3D9 2806 JRZ OBCP
 E3D8 23 INX H
 E3DC 7E MOV A,M
 E3D0 2B DCX H
 E3DE B9 CMP C
 E3DF 2006 JRNZ SKP
 E3E1 23 OBCP INX H ;READ NEXT BYTE
 E3E2 7E MOV A,M
 E3E3 2B DCX H ;DECH ADDRESS
 E3E4 CDE1E2 CALL ERR ;PRINT CODES
 E3E7 CD03E3 SKP CALL BMP ;CHECK IF DONE
 E3EA 20E5 JRNZ CONT ;BACK FOR MORE
 E3E8 F1 POP PSW
 E3ED C9 RET
 E3EE ; INPUT DATA FROM A PORT
 E3EE CD77E5 PINPT CALL PTSTNG
 E3F1 494EA0 DTII 'IN' ;READ 2 DIGITS
 E3F4 C0DDE1 CALL AIE2
 E3F7 4B MOV C,E
 E3F8 ED78 INP A
 E3FA C3EAE2 JMP PT2
 E3FD ; OUTPUT TO A PORT
 E3FD CD77E5 POUTP CALL PTSTNG
 E400 4F5554A0 DTII 'OUT' ;READ 2 DIGITS
 E404 C0DDE1 CALL AIE2 ;READ 2 DIGITS
 E407 C0DDE1 CALL AIE2
 E40A 4D MOV C,L
 E40B ED59 OUTP E
 E40C C9 RET
 E40E ;

```

E40E ; ***** VIDEO DRIVER FOR FLASHWRITER II *****
E40E ; CONTROL CODE COMMANDS:
E40E ; (B) HOME CURSOR
E40E ; (D) CLEAR SCREEN
E40E ; (E) PRINT CONTROL CODE
E40E ; (II) BACKSPACE
E40E ; (I) TAB
E40E ; (J) LINEFEED
E40E ; (M) CARRIAGE RETURN
E40E ; (N) NO CURSOR
E40E ; (P) CLEAR TO END OF SCREEN
E40E ; (Q) CLEAR TO END OF LINE
E40E ; (R) CURSOR DOWN
E40E ; (T) TOGGLE REVERSE VIDEO
E40E ; (U) CURSOR UP
E40E ; (W) CURSOR LEFT
E40E ; (X) CLEAR TO START OF LINE
E40E ; (Z) CURSOR RIGHT
E40E ; ESC XY POSITION LEAD-IN
E40E ; ***** VIDEO BOARD PARAMETERS *****
E40E 0050 = HORIZ EQU 80 ;NO. OF CHARACTERS
E40E 0018 = VERT EQU 24 ;NO. OF LINES
E40E ; 3E14 TVIDEO MVI A,'T'-64 ;TOGGLE VIDEO
E410 ; E410 F5 VIDEO PUSH PSW
E411 CS PUSH B
E412 DS PUSH D
E413 E5 PUSH H
E414 E67F ANI 07FH ;MASK OFF MSBIT
E416 4F MOV C,A ;PUT CHAR IN C
E417 CDECE4 DISPL CALL LIFTCURS ;ERASE CURSOR
E41A 3AEAFB LDA XYFLAG ;GET POSITIONING FLAG
E41D A7 ANA A ;CHECK IF TRUE
E41E 280A JNZ NOXY ;SKIP IF FALSE
E420 3D DCR A ;DECREMENT FLAG
E421 32EAFB STA XYFLAG ;SAVE NEW VALUE
E424 CA3CE5 JZ YPOS ;Y IF SECOND VALUE
E427 CJ2AES JMP XPOS ;ELSE X
E42A 79 NOXY MOV A,C ;RECOVER CHARACTER
E42B FE20 CPI ' ' ;PRINTING CODE?
E42D F261F4 JP PRINT
E430 FE1C CPI PCL-TABL ;TOO LARGE?
E432 F2CE84 JP RET
E435 E5 PUSH H ;CURSOR IN MEMORY
E436 2144E4 LXI H,TABL ;TABLE START

E439 5F MOV E,A
E43A 1600 MVI D,B
E43C 19 IAD D
E43D 5E MOV E,M
E43E 2160E4 LXI H,PCL
E441 19 DAD D
E442 E3 XTHL
E443 C9 RET
E444 ; CONTROL CHARACTER JUMP TABLE
TABL DB RET-PCL ;B
DB RET-PCL ;A
DB HOME-PCL ;B HOME CURSOR
DB RET-PCL ;C
DB FORM-PCL ;D CLEAR SCREEN
DB PCL-PCL ;E PRT CONTROL
DB RET-PCL ;F
DB RET-PCL ;G
DB DACKSP-PCL ;H BACKSPACE
DB TAB-PCL ;I TAB OVER
DB LINF-PCL ;J LINE FEED
DB RET-PCL ;K
DB RET-PCL ;L
DB CRET-PCL ;M CARRIAGE RET
DB RETB-PCL ;N NO CURSOR
DB RET-PCL ;O
DB CLEND-PCL ;P CLR SCR TO END
DB CLLINE-PCL ;Q CLR LINE TO END
DB LINF-PCL ;R CURSOR DOWN
DB RET-PCL ;S
DB TVIDF-PCL ;T TOGGLE VIDEO
DB CURSUP-PCL ;U CURSOR UP
DB RET-PCL ;V
DB BACKSP-PCL ;W CURSOR LEFT
DB CLSTART-PCL ;X CLR START OF LN
DB RET-PCL ;Y
DB EOL-PCL ;Z CURSOR RIGHT
DB LEDIN-PCL ;[ ESC=XY LEADIN

E460 ; PRINT CODE IN B REGARDLESS
E460 48 PCL MOV C,B
E461 ; PRINT THE CHARACTER ON THE SCREEN
E461 3A0XXFB PRINT LDA VFL
E464 A9 XRA C
E465 77 MOV H,A
E466 ; EOL CHECKS THE CURS POS FOR END OF LINE
E466 3AD8FB EOL LDA CURPOS
E469 3C INR A
E46A FE50 CPI HORIZ
E46C 305D JRC TABRET
E46E AF XRA A
E46F 32D8FB STA CURPOS
E472 ; MOVE IN 1 LINE
E472 3A0XXFB LINF LDA LINENO
E475 FE17 CPI VERT-1
E477 2023 JNZ2 HORIZ
E479 ; SCROLL UP ONE LINE
E479 21500H SCROLL LXI H,HORIZ

```

```

E47C ED5BDFFB
E480 19
E481 ED48 SCRL
E483 ED48
E485 7C
E486 FEF7 CPI HORIZ*VERTSCREEN/256
E488 20F7 JRNZ SCRL
E48A 7D MOV A,H
E48B FE80 CPI HORIZ*VERT+SCREEN&OFFH
E48D 20F2 JRNZ SCRL
E48F 3ADCFB LDA LINENO
E492 ; ERASE BOTTOM LINE
E492 EB EBOTL XCIG
E493 0650 MVI B,HORIZ
E495 3620 ELOP MVI M,'
E497 23 INX H
E498 05 DCR B
E499 20FA JRNZ ELOP
E49B 3D DCR A
E49C 3C NOSCRL INR A
E49D 32DCFB STA LINENO
E4A0 182C JR RET
E4A2 ; ERASE BEFORE BACKSPACING
E4A2 3620 DBACKSP MVI M,20H
E4A4 3ADBFU LDA CURPOS
E4A7 A7 ANA A
E4A8 2824 JRNZ RET
E4A9 3D DCR A
E4AB 2B DCX H
E4AC 3620 MVI M,'
E4AE 181B JR TABRET
E4B0 ; MOVE THE CURSOR BACK
E4B0 3ADBF8 BACKSP LDA CURPOS
E4B3 3D DCR A
E4B4 F2CB84 JP TABRET
E4B7 1811 JR CRET
E4B9 ; TAB OVER TO THE NEXT 8 MULTIPLE
E4B9 3ADBF8 TAB LDA CURPOS
E4C0 F687 ORI 7
E4B8 18A9 JR EOL+3
E4C0 ; CLEAR THE SCREEN AND HOME UP
E4C0 CD10E5 F0H CALL CLEAR
E4C1 AF HOME XRA A
E4C4 32DCFB STA LINENO
E4C7 32D0FB STA VFL ; CLR VID FLAG
E4CA ; CARRIAGE RETURN
E4CA AF CRET XRA A
E4CB 32D0FB TABRET STA CURPOS
E4CE ; RETURN TO THE CALLING ROUTINE
E4CE CD0CE4 RET CALL LIFTCURS
E4D1 E1 POP H
E4D2 D1 POP D
E4D3 C1 POP B
E4D4 F1 POP PSW
E4D5 C9 RET
E4D6 3AD0FB TVTDE LDA VFL

```

```

E4D9 EE80 XRI 80H
E4D8 32D0FB STA VFL
E4D8 18EE JR RET
E4E0 ; MOVE THE CURSOR UP
E4E0 CURSUP LDA LINENO
E4E1 A7 ANA A
E4E4 20E8 JRZ RET
E4E6 3D DCR A
E4E7 32DCFB STORLN STA LINENO
E4E8 18E2 JR RET
E4E9 ; CALCULATE MEM ADD FROM CURSOR POSITION
E4E9 3ADCFB LIFTCURS LDA LINENO ; GET CURRENT LINE
E4E9 CD67B5 CALL CALCULATELINE ; GET MEMORY ADDRESS FOR LINE
E4F2 ED5B00FB LOED CURPOS ; GET POSITION ON LINE
E4F6 1600 MVI D,B ; DB = X POSITION
E4F8 19 DAD D ; HL = CURSOR POSITION
E4F9 7E MOV A,H ; GET CHARACTER
E4FA EE80 XRI 80H ; REVERSE VIDEO
E4FC 77 MOV M,A ; PUT CHAR BACK
E4FD C9 RET
E4FE ; CLEAR TO END OF SCREEN
E4FE ; CLEAR TO END OF LINE
E4FE CD1AE5 CLEND CALL WRSPC
E501 18CB JR RET
E503 3AD8FB CLINE LDA CURPOS
E503 3AD8FB CLINE LDA CURPOS
E506 3620 MVI M,'
E508 23 INX H
E509 3C INR A
E50A FE50 CPI HORIZ
E50C 20F8 JRNZ CLINE+3
E50E 18B8 JR RET
E510 ; CLEAR THE SCREEN
E510 2100F0 CLEAR LXI H,SCREEN
E513 22DFFB SHLD TOGN ; SET TOP OF SCREEN
E516 AF XRA A
E517 32EAFB STA XYFLAG ; ZERO XY LEAD-IN FLAG
E518 3620 WRSPC MVI M,'
E51C 23 INX H
E51D 7C MOV A,H
E51E FEF8 CPI SCREEN+2048/256
E520 20F8 JRNZ WRSPC
E522 C9 RET
E523 ; PROCESS LEAD IN CODE
E523 3E02 LEDIN MVI A,2
E525 32EAFB STA XYFLAG
E528 18A4 JR RET
E52A ; SET X AND Y CURSOR POSITIONS
E52A 79 XPOS MOV A,C ; GET X POSITION
E52B FE7F CPI 7FH ; CHECK FOR 07FH
E52D 2005 JRNZ XPOS10 ; SKIP IF NOT
E52F 32EAFB STA TOPFLAG ; SET TOPFLAG

```

```

E532 189A
E534 FE50
E536 3893
E538 3E4F
E53A 188F
E53C
E53C 3A02FB
E53F B7
E540 2012
E542 79
E543 FE18
E545 38A8
E547 3E17
E549 189C
E54B
E54B AF
E54C 32D0FB
E54F D2CEE4
E552 18AF
E554
E554 1 SET TOP OF SCREEN TO LINE SPECIFIED BY A
E554
E554 AF
TOPSCREEN: XRA A
E555 32EDFB
E558 79
E559 FE17
E55B D2CEE4
E55E CD67E5
E561 22DFFB
E564 C3CEE4
E567
E567 1 LINE ADDRESS CALCULATION (LINE IN A) RETURNS ADDRESS IN HL
E567 1 OPTIMIZED AT BOTTOM
E567
E567 2180F7
E56A 11B0FF
E56D 3C
CALCLOOP: INR A
E56E 19
DAD D
E56F FE18
E571 20FA
E573 C9
RET
E574
E574 1 PRINT A STRING
E574
E574 CD02E1
PTSTRING CALL CRLF ;CRLF FIRST
E577 E3
PTSTRING XTHL ;GET STRING POINTER
E578 76
MOV A,M ;GET CHAR
E579 23
INX H ;INCR POINTER
E57A E3
XTHL ;PUT POINTER BACK
E57B A7
ANA A ;ZERO CARRY
E57C CD10E4
CALL VIDEO ;PRINT IT
E57F F8
IM ;RETURN IF NEGATIVE
E580 18F5
JR PTSTRING ;REPEAT IF NOT
E582
E582 CD74E5
PROMPT CALL PTSTRING
DTII 'Exec> '

```

```

E588 C9
E58C
E58C 7E ;WMP2
RET
MOV A,M
MOV B,A
MVI A,'E'-64
CALL VIDEO
CALL BMP
R2
DCR C
IM
JR WMP2
E590 ; HOME CURSOR, PRINT "ADDR"
HOME CALL RPTRSTC
E592 14 DB 'T'-64
E59P 41444452 DTH 'ADDR '
E5A3 A0
E5A4 0680 MVI B,0
E5A6 3E18 MVI A,24
E5A8 32D0FB STA WIDMII
E5A8 C9 RET
E5A9 ; MAKE A RULER FOR HEX DUMP
HEXRULER MOV A,B
E5AD FE10 CPI 16
E5A9 2806 JRZ HEXRCT
E5B1 CD2B27 CALL PT2S
E5B4 04 INR B
E5B5 18F5 JR HEXRULER
E5B7 ; EXTEND FOR ASCII
HEXRCT CALL SPCE
E5B7 CD40E1 CALL SPCE
E5B8 CD40E1
E5B9 0680 MVI B,0
E5B9 78 HEXRLP MOV A,B
E5C0 FE10 CPI 16
E5C2 C8 R2
E5C3 E68P ANI 0FII
E5C5 CDF5E2 CALL BINL
E5C8 04 INR B
E5C9 18F4 JR HEXRLP
E5CB ; HEX DUMP ROUTINE
E5CB CD77E5 CALL PTSTRING
E5CE 44554D50 DTII 'DUMP '
E5D2 A0
E5D3 CD41E1 CALL TAHEX
E5D6 CD90E5 CALL HOME
E5D9 CD40E5 CALL HEXRULER
E5DC CD0EE4 CALL TVIDEO
E5DF CD03E6 CALL SETSCRL
E5E2 CD03E2 HLP1 CALL PTAD
E5E5 E5 PUSH H
E5E6 D5 PUSH D
E5E7 0E10 MVI C,16
E5E9 7E HLP2 MOV A,M
E5EA CD2B27 CALL PT2S
E5E9 23 INR H
E5E9 8D DCR C
E5E9 C2E9E5 JRZ HLP2
E5F2 D1 POP D

```

```

E5F3 E1      POP    H
E5F4 0E0P    MVI    C,15
E5F6 CDADE1  CALL   SPC1
E5F9 CDADE1  CALL   SPC2
E5FC CD8CE5  CALL   WDMP2
E5FF FADFE5  JM    HLP1-3
E602 C9      RET
E603 ; CHECK TO SET SCROLL POINT
E603 3ADEFB  SETSCRLL LDA    WIDTH
E606 3D      DCR    A
E607 32DEFB  STA    WIDTH
E60A 2007  JRNZ  CTECRL
E60C 8150F0  LXI    B,SCREEN+50H ;2ND LINE
E60F ED43DFB  SBCD   TOSN   ;SCROLL POINT
E613 C9      CTSCRL RET
E614 ; PROGRAM MEMORY
E614 CD77E5  PROGRAM CALL   PTSTNG
E617 50524F47 DTH    'PROGRAM'
E618 52414DA0
E61F CD92E1  CALL   AHEX   ;ADDR IN HL
E622 ED53E1FB SED    TCURPOS
E626 CD9BES  CALL   IOMEc
E629 CD8CE5  CALL   HEXRULER
E62C CD8EE4  CALL   TVIDEO
E62F AF      XRA    A
E630 32DEFB  STA    WIDTH
E633 CD9D66  CALL   PRTILINE ;PRINT LINE CONT H
E636 CD82E2  POLLOOP CALL   ESCAPE
E639 CDC0E1  CALL   IHEX
E63C 2AE1FB  LHLD   TCURPOS
E63F 301A  JRNC  MO0HMEM
E641 ; CONTROL CODE TABLE
E641 FE20  CPI    ''
E643 2846  JRZ    CSRT
E645 FE08  CPI    8
E647 2845  JRZ    CSLT
E649 FE12  CPI    'R'-64
E64B 2839  JRZ    CS0N
E64D FE15  CPI    'U'-64
E64F 282F  JRZ    CSUP
E651 FE17  CPI    'W'-64
E653 2839  JRZ    CSLT
E655 FE1A  CPI    'Z'-64
E657 2832  JRZ    CSRT
E659 18D8  JR    POLLOOP
E65B ; MODIFY A MEMORY LOCATION
E65B 2AE1FB MO0HMEM LHLD   TCURPOS
E65E 4F      MOV    C,A
E65F 3ADEFB LDA    WIDTH
E662 A7      ANA    A
E663 7E      MOV    A,M
E664 280D  JRZ    LSNIBL
E666 E6F0  ANI    0F0H
E668 B1      ORA    C
E669 77      MOV    H,A
E66A 3ADEFB REMEM LDA    WIDTH

```

```

E66D EE01  LSNIBL
E66F 281F  JRNZ  1
E671 1818  JR    CSRT
E673 17      RAL
E674 17      RAL
E675 17      RAL
E676 17      RAL
E677 E6F0  ANI    0F0H
E679 81      ORA    C
E67A 0F      RNC
E67B 0F      RNC
E67C 0F      RNC
E67D 0F      RNC
E67E 18E9  JR    REMEM
E680 ; MOVE UP ONE LINE
E680 11F0FF  CSUP  LXI    D,-16
E683 19      DAD
E684 1809  JR    RTRN
E686 ; MOVE DOWN ONE LINE
E686 111000  CS0N  LXI    D,16
E689 18F8  JR    CSUP+3
E68B ; MOVE RIGHT ONE SPACE
E68B 23      CSRT  INX    H
E68C 1801  JR    RTRN
E68E ; MOVE LEFT ONE SPACE
E68E 2B      CSLT  DCX    H
E68F I
E69F AP      RTRN  XRA  A
E690 32DEFB  STA    WIDTH
E693 22E1FB  SHLD  TCURPOS
E696 3E15  UPAROW MVI  A,'U'-64
E698 CD10E4  CALL   VIDEO
E698 1896  JR    POLLOOP-3
E69D ; PRINT A LINE CONTAINING ((H))
E69D 2AE1FB  PRTILINE LHLD   TCURPOS
E6A0 E5      PUSH  H
E6A1 D1      POP    D
E6A2 7D      MOV    A,L
E6A3 F60F  ORI    0F11
E6A5 5F      MOV    E,A
E6A6 E6F0  ANI    0F0H
E6A8 6F      MOV    L,A
E6A9 CD82E5  CALL   HLP1
E6AC ; NOW PUT CURSOR WHERE IT GOES
E6AC CD8CE4  CALL   LIFTCURS
E6AF 2AE1FB  LHLD   TCURPOS
E6B2 7D      MOV    A,L
E6B3 E60F  ANI    0F11
E6B5 6F      MOV    L,A
E6B6 3E05  MVI    A,5
E6B8 2D      PLOPI  DCR  L
E6B9 FAC0E6  JM    PCOUNT
E6BC C603  ADI    3
E6BE 18F8  JR    PLOPI
E6C0 6F      PCOUNT  MOV    L,A
E6C1 3ADEFB  LDA    WIDTH
E6C4 85      ADD

```

```

E6C5 ; A = 5+3*LHW
E6C5 J2D8FB STA CURPOS
E6C8 C3ECE4 JMP LIFTCURS
E6CB ;
E6CB ;
E6CB ; DISPLAY REGISTERS
E6CB CD77E5 DRS CALL PTSTNG
E6CE 52454749 DRS CALL 'REGISTERS'
E6D2 53544552
E6D6 D3
E6D7 ; DUMP REGISTERS AFTER ENTRY FROM RST 7
E6D7 E3 DUMPREGS XTHL
E6D8 F5 PUSH PSW
E6D9 CD31E7 CALL DISPREGS
E6D9 2B DCX H ;GET BREAK ADD
E6D9 CDD3E2 CALL PTAD
E6E0 E1 POP H
E6E1 C5 PUSH B
E6E2 CD86E7 CALL PRTFLGS
E6E5 C1 POP B
E6E6 CDD6E2 CALL PTAD+3 ;PRINT AF
E6E9 E1 POP H
E6EA 22EJFB SILD HLTEMP
E6ED CDA7E7 CALL PTIREE ;PRINT B D H
E6F0 DDE5 PUSH IX
E6F2 E1 POP H
E6F3 CDD6E2 CALL PTAD+3 ;PRINT IX
E6F6 FDE5 PUSH IY
E6F8 E1 POP H
E6F9 CDD6E2 CALL PTAD+3 ;PRINT IY
E6FC 210000 LXI H,0
E6FF 39 DAD SP
E700 22E5FB SILD SPTEMP
E703 CDD6E2 CALL PTAD+3 ;PRINT SP
E706 88 EXAF
E707 F5 PUSH PSW
E708 E1 POP H
E709 CDD6E2 CALL PTAD+3
E70C D9 EXX
E70D CDA7E7 CALL PTIREE
E710 D9 EXX
E711 8A LDAX B
E712 CD2BE7 CALL PF2S
E715 1A LDAX D
E716 CD2BE7 CALL PT2S
E719 2AE3FB LILD HLTEMP
E71C 7E MOV A,M
E71D CD2BE7 CALL PF2S
E720 2AE5FB LILD SPTEMP
E723 F9 SPHL
E724 E1 POP H
E725 CDD6E2 CALL PTAD+3
E728 C30B80 JMP CLRBRK ;CLEAR BREAKPOINT
E72D CDEAE2 PF2S CALI. PT2
E72E C3A0E1 JMP SPCE ;PRINT 2 CHARS
E731 ; DISPLAY REGISTER HEADER ON SCREEN

```

```

E731 CD74E5 DISPREGS CALL RPTING
E734 14 DD 'T'-64
E735 41444452 DT 'ADIR FLAGS AF BC DE'
E739 20464C41
E740 47532020
E741 41462020
E745 20424320
E749 20204445
E74D 20202048 DT ' HL IX IY SP '
E751 4C202020
E755 49582020
E759 20495920
E75D 20205350
E761 20
E762 20204146 DT ' AF '
E766 27 DB 27H
E767 20204243 DT ' BC '
E768 27 DB 27H
E76C 20204445 DT ' DE '
E770 27 DB 27H
E771 2020484C DT ' HL '
E775 27 DB 27H
E776 20404220 DT ' BB BD BH BSP '
E77A 40442040
E77E 48204053
E782 5020
E784 94 DB 'T'+64
E785 C9 RET
E786 ; PRINT FLAGS
E786 015A40 PRTFLGs LXI B,405AII
E789 CD86E7 CALL MASKFLG
E78C 014301 LXI B,143II
E78F CDB6E7 CALL MASKFLG
E792 014D00 LXI B,04DII
E795 CDB6E7 CALL MASKFLG
E798 014504 LXI B,445II
E79D CDB6E7 CALL MASKFLG
E79E 014810 LXI B,1048II
E7A1 CDB6E7 CALL MASKFLG
E7A4 C3A0E1 JMP SPCE
E7A7 ; PRINT BC DE HL IN ORDER
E7A7 E5 PTIREE PUSH H
E7A8 C5 PUSH B
E7A9 E1 POP H
E7AA CDD6E2 CALL PTAD+3
E7AD D5 PUSH D
E7AE E1 POP H
E7AF CDD6E2 CALL PTAD+3
E7B2 E1 POP H
E7B3 C3D6E2 JMP PTAD+3
E7B6 ; MASKPIG
E7B6 7D MOV A,L
E7B7 A8 ANA B
E7B8 3E20 MVI A,20H
E7B9 CA10E4 JZ VIDEO

```

```

E7BD 79          MOV  A,C
E7BE C310E4      JMP  VIDEO
E7C1
E7C1  /          SET BREAKPOINT
E7C1
E7C1 CD77E5      SETBRK   CALL  PTSTNG
E7C4 42524541      DTH    'BREAK AT '
E7C8 48204154
E7CC A0
E7CD CD92E1      CALL  ANEX
E7D0 1A          LDAX  'D
E7D1 32E9FB      STA   BRKCODE
E7D4 ED53E7FB      SDED   BKPTLOC
E7D8 3EEF          HVI   A,BEEH  /RESTART 5
E7DA 12          STAX  D
E7DB C9          RET
E7DC
E7DC  /          EXTERNAL COMMUNICATIONS
E7DC CD77E5      EXTCOM  CALL  PTSTNG
E7D0 45585420      DTH    'EXT COM '
E7E3 434F41A8
E7E7 D005          RECEIVE  IN    5
E7E9 E602          NH1    2
E7E8 2805          JRZ    NEXCHR
E7ED D004
E7EF CD10E4      CALL  VIDEO
E7F2 CD02E2      NEXCHR  CALL  ESCAPE
E7F5 28F0          JRZ    RECEIVE
E7F7 D304          OUT    4
E7F9 18EC          JR    RECEIVE
E7FB
E7FB          PRT   'PROGRAM LENGTH = ',$-BEGIN+1
E7FB          ORG   BASE+7FH
E7FF 50          VERSION: DB    50H
E800
E800  /          CURSOR STORAGE LOCATIONS
E800
E800
E800  /          ORG   SPTR+0BH
FB08  CURPOS  DS    1          /POS ON LINE
FB0C  LINENO   DS    1          /LINE NUMBER
FB0D  VPL     DS    1          /REVERSE VID FLAG
FB0E  WIDTHH   DS    1          /PRINT WIDTH
FB0F  TMON    DS    2          /TOP OF SCREEN
FBE1  TMONPOS  DS    2          /TEMP POSITION
FBE3
FBE3  /          TEMPORARY STORAGE LOCATIONS FOR REGISTERS, ETC.
FBE3
FBE3  HITEMP  DS    2
FBE5  SPTMP   DS    2
FBE7  BKPTLOC: DS    2          /BREAKPT LOCATION
FBE9  BRKCODE DS    1          /CODE AT BREAKPT
FBEA  XYFLAG  DS    1          /CURSOR XY FLAG
FBEA  TOPFLAG  DS    1          /NON-ZERO IF TOPSCREEN SET

```